

HTML5 para Juniors – Bases sólidas para un desarrollo profesional

Guía clara para escribir HTML correcto, semántico y profesional

© 2024 **CONECTA-JR** | Desarrollado por **Vanesa Paola Soria** | Todos los Derechos Reservados

Material educativo de libre distribución para la comunidad Junior.



Introducción



HTML5 no es simplemente una herramienta para "maquetar" o dar una apariencia visual básica; es la columna vertebral de cualquier proyecto web profesional. Su rol fundamental es permitir que el navegador examine el código y construya el DOM (Document Object Model), que es la estructura de nodos sobre la cual descansan los estilos y la interactividad.

Sigue siendo clave hoy porque una estructura sólida facilita que el CSS y el JavaScript interactúen con el contenido de forma predecible y eficiente.

¿Qué es HTML5?

HTML5 es un lenguaje de marcado, lo que significa que utiliza etiquetas específicas para identificar y organizar los elementos de un documento, pero **no es un lenguaje de programación**.

Estructura (HTML)

Define qué es cada parte del contenido (un párrafo, un encabezado, un formulario).

Estilos (CSS)

Se encarga de la presentación y el diseño visual de esos elementos.

Comportamiento (JavaScript)

Permite modificar los estilos y la lógica del documento en tiempo real.

- ❏ **Semántica:** Es la práctica de dar un significado claro al contenido, permitiendo que tanto los navegadores como los desarrolladores entiendan la función de cada etiqueta sin necesidad de ver el diseño.

Estructura base de un documento HTML profesional

Todo documento profesional debe comenzar con una estructura estandarizada que garantice la compatibilidad:

01

Doctype

`<!DOCTYPE html>` informa al navegador que está procesando un documento HTML5 moderno.

02

Etiqueta `<html>`

Contenedor raíz que debe incluir el atributo `lang` (ej. `lang="es"`) para indicar el idioma, lo que mejora la accesibilidad y el SEO.

03

Etiqueta `<head>`

Contiene metadatos no visibles, como el `meta charset="utf-8"` (vital para representar correctamente caracteres especiales) y el `title` del proyecto.

04

Etiqueta `<body>`

Donde reside todo el contenido que el usuario final verá en el navegador.

Esta base impacta directamente en cómo los buscadores (SEO) indexan la página y cómo las herramientas de accesibilidad interpretan la información para usuarios con discapacidades.

Etiquetas fundamentales y HTML semántico

Etiquetas fundamentales que todo Junior debe dominar

◆ Texto y jerarquía

- **Encabezados** (`<h1>` a `<h6>`): Se utilizan para establecer la importancia del contenido. El `<h1>` es el título principal y debe haber una jerarquía lógica descendente.
- **Párrafos** (`<p>`): La etiqueta estándar para bloques de texto.

◆ Listas

- `` (**Lista desordenada**): Para elementos sin un orden específico, usualmente representados con viñetas.
- `` (**Lista ordenada**): Para pasos o elementos que requieren numeración.
- ``: Representa cada ítem individual dentro de una lista.

◆ Enlaces

- **Etiqueta** `<a>`: Define hipervínculos utilizando el atributo `href`.
- **Rutas**: Pueden ser internas (vínculos a archivos dentro del mismo proyecto) o externas (URLs completas a otros sitios).

Contenedores y estructura

El elemento `<div>` es un contenedor de nivel de bloque que ocupa todo el ancho disponible de su padre. Aunque es muy versátil para agrupar elementos, carece de significado semántico y su uso excesivo ("divitis") es una limitación en el desarrollo moderno. **Un desarrollador profesional entiende que la estructura importa más que el diseño, ya que un HTML bien organizado permite que el CSS sea más mantenible y el sitio más accesible.**

HTML semántico (nivel profesional)

El HTML semántico utiliza etiquetas que describen su propósito. **Esto es lo que los Seniors y Recruiters valoran, ya que demuestra un nivel profesional superior.**

`<header>`

Cabecera de la página o de una sección.

`<nav>`

Contenedor para los enlaces de navegación principal.

`<section>`

Agrupar contenido temático relacionado.

`<article>`

Para contenido independiente y autónomo (como un post de blog).

`<main>`

Define el contenido principal y único del documento.

`<footer>`

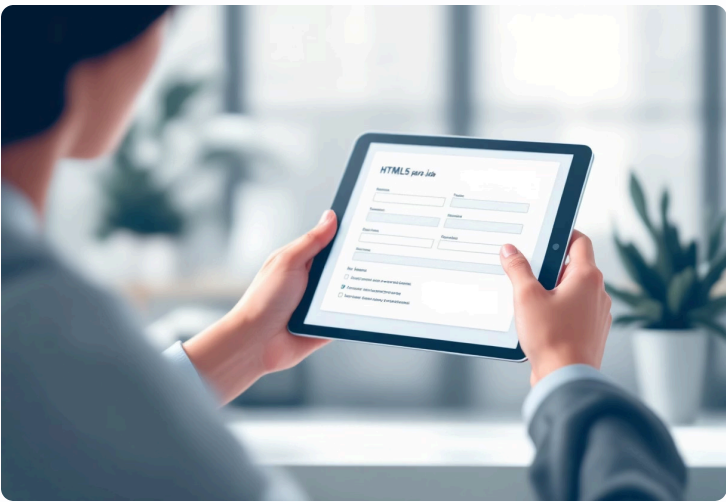
Pie de página con información de contacto o derechos de autor.

Formularios, buenas prácticas y próximos pasos

Formularios y elementos importantes

Los formularios son el punto crítico de la Experiencia de Usuario (UX), ya que permiten la interacción directa.

- **Estructura básica:** Se definen con la etiqueta `<form>`.
- **Inputs comunes:** `text`, `password`, `email`, `number`, y `range`.
- **Importancia del `<label>`:** Siempre debe usarse junto a un input (mediante el atributo `for`) para que el usuario sepa qué información debe ingresar y para mejorar la accesibilidad.
- **Validaciones:** HTML5 permite validaciones básicas mediante atributos como el `type` de input (ej. `type="email"`), asegurando que los datos sigan el formato correcto antes de ser enviados.



Buenas prácticas en HTML5

Escribir un código prolijo no solo es estética; es profesionalismo:

- ### Código legible

Mantener una distinción clara entre contenido y presentación.
- ### Uso de Clases e IDs

Los IDs deben ser únicos en toda la página; las clases se usan para estilos que se repiten en múltiples elementos.
- ### Comentarios

Utilízalos para explicar secciones complejas o marcar el inicio y fin de componentes grandes.
- ### Accesibilidad

Asegúrate de que el contenido sea navegable y comprensible para todos los usuarios.

Qué sigue después de HTML5

Una vez dominada la estructura, el camino profesional continúa con:

- ### CSS

Para el diseño, estilos y maquetación avanzada con Flexbox y Grid.
- ### JavaScript

Para añadir interactividad y manipular el contenido dinámicamente.
- ### Responsive Design

Adaptar tus sitios a cualquier tamaño de pantalla mediante media queries.
- ### Documentación oficial

Acostúmbrate a consultar fuentes técnicas y notas para profesionales para mantenerte actualizado.

←

END

Cierre

Un Junior profesional no es quien hace que un sitio "se vea bien", sino quien construye una base estructural sólida, semántica y accesible. Dominar estas bases es lo que te permitirá escalar a tecnologías más complejas con éxito.

NOTA: No te detengas en la teoría. Practica, construye proyectos reales y mantén un aprendizaje constante. La web está en evolución permanente y tu código debe hacerlo con ella.